Cardea: Dynamic Access Control in Distributed Systems

Rebekah Lepro.
NASA Advanced Supercomputing (NAS) Division
NASA Ames Research Center, Moffett Field, CA 94035
rlepro@arc.nasa.gov

NAS Technical Report NAS-03-020
November 2003

Abstract

Modern authorization systems span domains of administration, rely on many different authentication sources, and manage complex attributes as part of the authorization process. This paper presents Cardea, a distributed system that facilitates dynamic access control, as a valuable piece of an inter-operable authorization framework. First, the authorization model employed in Cardea and its functionality goals are examined. Next, critical features of the system architecture and its handling of the authorization process are then examined. Then the SAML and XACML standards, as incorporated into the system, are analyzed. Finally, the future directions of this project are outlined and connection points with general components of an authorization system are highlighted.

1. Introduction

The environment framing the modern authorization process span domains of administration, relies on many different authentication sources, and manages complex attributes as part of the authorization process. Cardea facilitates dynamic access control within this environment as a central function of an inter-operable authorization framework. The system departs from the traditional authorization model by separating the authentication and authorization processes, distributing the responsibility for authorization data and allowing collaborating domains to retain control over their implementation mechanisms. Critical features of the system architecture and its handling of the authorization process differentiate the system from existing authorization components by addressing common needs not adequately addressed by existing systems. Continuing system research seeks to enhance the implementation of the current authorization model employed in Cardea, increase the robustness of current features, further the framework for establishing trust and promote interoperability with existing security mechanisms.

2. Background

2.1. Framing the authorization problem

Authorization is process by which a party can gain access to specific information or functionality. Administrators can protect the resources within their domain by indicating what constitutes appropriate access for each party that may access a particular resource. Granting authorization usually requires an evaluation of an authenticated identity to determine the rights or permissions that were previously assigned to that identity. Historically, controlling access to protected resources has been predominantly implemented with resource specific mechanisms. As resource interaction further dominates computing goals, these mechanisms must interoperate to determine and apply appropriate authorization decisions. As systems gravitate from a centralized to a distributed model, this interoperability needs remodel existing requirements and place new requirements on authorization.

Authorization in an inherently distributed system requires widespread cooperation between separate and autonomous administrative domains [FOS01]. Maintaining a consistent authorization strategy across participants requires each system to maintain at least some knowledge of its potential collaborators throughout the entire system. Further, each authorization decision that spans two or more authorization domains requires each participant in the decision be capable of correctly producing, accepting and interpreting authorization information from a group of potentially heterogeneous peers. This capability requires joint agreement on protocol, syntax and semantics for each piece of shared authorization data. Additionally, existing enforcement mechanisms typically associate authorization data, be it individual, group, or role based, with identities that are unique to an individual authorization domain [MIR01]. This requires a level of synchronization of local identities between the domains, which ultimately forces administrative domains to cede partial control of local authorizations to a literal or figurative central authority.

2.2. Traditional authorization solutions

As systems evolve, the need to control access to resources that belong to that system increases. Regardless of system complexity, authorization begins with an offline evaluation of each protected resource to determine requirements that must be satisfied for access to be granted to that resource [MUD01]. Traditionally, a long-term local identity represented a unique set of permissions to a single resource [HUM01]. Authorizing a request required the requestor to specify the correct local identity embodying the necessary local rights to complete an action and provide a security token, such as a password, to confirm the requestor could assume the specified identity. With this model, authorizing each new user required the creation of a new local identity to represent the permissions particular to the new user. Further, each new identity must be created and configured before any access is granted via the rights associated with it. Therefore, each resource administrator must know, a priori, each potential user of the resource, the appropriated rights to grant that user and link the permissions to the correct local identity. Obviously, the volume of authorization data to manage as well as the complexity of that data escalates with the addition of each new resource or user.

Group authorization leverages the overlap among users authorizations to simplify management of controlled resources. Rather than define authorization in terms of individual identities and access rights, unique authorizations are assigned to a group, to which individual identities are then assigned. Thus, each grouping of users logically represents one or more common characteristics possessed by each member. Each authorization decision automatically considers rights assigned to each assigned such that group members may act directly with group authority of group rather than through individually assigned rights. This approach adds a level of indirection into the authorization process by logically grouping similar permissions and users thereby reducing the volume of authorization information to be managed. Although group authorization can significantly reduce the total volume of authorization data, it can only provide a common definition for identities, group or permission within an administrative domain. Further, group authorizations only relate users to a common permission. The correct set of groups must still be assigned for each potential user. Thus, the requirement to uniquely identify each potential user within a single administrative domain remains.

As users request access to external resources, each domain must individually establish appropriate groups and identities to manage access to its resources. As there is no correlation between the local identities in separate administrative domains, each domain must execute an authorization process regardless of whether the authorizations have already be granted in another domain. Further, executing appropriate authorization decisions within a local domain for an external user requires a common awareness of authorization data between the two participating domains. Thus, each domain must agree either with a common representation for authorization data or trust common super-domain that is an authoritative source for the data.

2.3. Standards that address the authorization process

In efforts to achieve trusted and common representations for authorization data, in support of distributed authorization and related problems, several standards are currently maintained or are under development by both the World Wide Web Consortium [W3C] and the Security Services Technical Committee of OASIS [OASIS]. These standards include the Security Assertion Markup Language (SAML) [SAML], the eXtensible Access Control Markup Language (XACML)[XACML] and XML Digital Signature Recommendation (XMLDSig) [XMLD]. Each of these standards defines a common language in XML for representing authorization data and providing framework support for transactions related to that data.

2.3.1. SAML

SAML, ratified by OASIS represents authorization information in the form of assertions pertaining to authentication acts, subject attributes and authorization decisions. The SAML standard defines a protocol by which a client requests assertions from a specific authority and receives a response containing the appropriate assertions. As SAML only defines how authorization information is communicated between entities, it imposes no changes on internal security architectures. [FAWC] Therefore, distinct administrative domains can maintain appropriate internal mechanisms and still effectively communicate with other domains to distribute the authorization process.

Three components work together to provide mechanisms through which SAML supports the authorization process. Core SAML syntax is expressed as XML [XML] constructs. The generic assertion construct contains data general to any assertion type such as identifier or version information. This general structure then contains a set of nested inner elements that contain data specific to a particular assertion type, such as subject information. Each SAML exchange consists of a request and a response. Additional XML constructs represent a SAML request or response. These constructs contain exchange specific information such as origin or destination information, queries or the assertions returned from those queries. Bindings for the language express how to map a single SAML exchange to a standard communication protocol. Profiles define how SAML assertions are embedded in objects that are communicated between producers and consumers of assertion data.

2.3.2. XACML

XACML, ratified by OASIS, provides a general-purpose mechanism for expressing the access control policies for an organization. XACML is designed to enable the interoperability of a broad range of administration and authorization products by providing a universal language for authorization policy. [WS01] First, XACML establishes a vocabulary for expressing the rules that define such policies. Then, XACML, similarly to SAML, defines a request/response protocol by which requests for access are made and the appropriate response determined. Each authorization decision reached by an XACML system relies on characteristics of the request subjects rather than on local identities that represent those characteristics. Conceptually, XACML defines

two components that work together during the authorization decision process: the policy enforcement point (PEP), which enforces a policy decision and the policy decision point (PDP), which evaluates a request and determines the appropriate authorization to grant. The PEP includes, in a standard manner, the necessary attributes within each authorization request to be considered during the authorization decision. XACML makes no requirement on either the initial format of attribute data or the method by which the PEP obtains the appropriate data. Instead, XACML presumes that all attributes will be presented in a standard manner to a PDP by the PEP so they may be evaluated in a consistent manner.

Each policy defined with XACML syntax is comprised of XML constructs that define individual rules representing the basic unit of management and which are combined, via a specified combining algorithm, into a policy. Each rule may evaluate characteristics of the requester, the requested resource, the desired action or the current environment in terms of well-known data-types and functions. XACML predefines a wealth of common types for use within these rules, such as string, date, or set; a group of commonly used operators, such as equality, or set comparison; and mechanisms to extend the basic definitions for the unique needs of a particular policy. XACML also defines a number of rule-combining algorithms that govern how individual rules within a single policy are evaluated together. Finally, XACML defines a method by which a policy may reference other policies. Thus it provides a way to distribute responsibility for each rule to the appropriate organization. With these features, XACML can represent virtually any access control policy need intrinsically or leverage the syntactic and semantic flexibility of XML to extend the core definition as necessary.

2.3.3. XMLDSig

The XML Digital Signature Recommendation specifies syntax and processing rules for providing integrity, messaging authentication and or signer authentication in an XML context. The above capabilities are provided by application of an XML signature to a digested version of arbitrary XML content. Before information is digested, one or more transformations to support functions like compression or canonicalization of the original data is applied. Then the digest, together with supporting information, is then cryptographically signed. XMLSig supports three distinct types of signatures: enveloped, enveloping and attached. With an enveloped XML signature, an XML construct containing all the information pertinent to the signing, such as the signature, signer identity or URI references to the signed data, becomes a child element the root of the signed XML document. With an enveloping XML signature, an XML construct containing all the information pertinent to the signing, such as the signature, signer identity or URI references to the signed data, encapsulates the signed XML document. With a detached XML signature, the XML construct containing all the information pertinent to the signing, such as the signature, signer identity or URI references to the signed data remains independent from the signed XML document. With all signatures,

the digest applies only to the actual signed content. When using an enveloped signature, the enveloped signature canonicalization transform excludes the signature element and its content from the digest generation.

XMLDSig provides the ability to sign specific portions of an XML construct. This allows different entities to sign distinct portions of a single document. Again, canonicalization transforms exclude XML signature data from signed content to ensuring integrity throughout the signing process. Therefore, the signature detail content relative to the first signature does not impact the actual signed content for any subsequent signature assigned. Further, identical content generates the signed digest for each signature if multiple signatures are applied to the same portion of a document. If distinct portions of an XML document are signed, changes affect only the signatures on that specific portion of the data instead of all signatures.

3. Cardea

Cardea is a distributed authorization system, developed as part of the NASA Information Power Grid [FOS02, FOS03, IPG], which dynamically evaluates authorization requests according to a set of relevant characteristics of the resource and requester rather than considering specific local identities. Potentially accessed resources within an administrative domain are protected by local access control policies, specified with the XACML syntax, in terms of requester and resource characteristics. Further, potential users are identified by X.509 proxy certificates [RFC2459, TUE01] but only modeled according to the characteristics they can reliably demonstrate. The exact information needed to complete an authorization decision is assessed and collected during the decision process itself. This information is assembled appropriately and presented to the PDP that returns the final authorization decision for the actual access request together with any relevant details. See figure 1 for a system architecture illustration.

Cardea is currently implemented in the Java [JAVA] language as a set of independent components. Conceptually, the system contains a SAML Policy Decision Point (SAML PDP), one or more Attribute Authorities (AA), one or more Policy Enforcement Points (PEP), one or more references to an Information Service (IAS), an XACML context handler, one or more XACML Policy Administration Points (PAP) and an XACML Policy Decision Point (XACML PDP). Although all these components may be co-located on the same machine to use local communication paradigms, they may also be distributed across several machines and their functionality exposed as web service portTypes [WSDL].

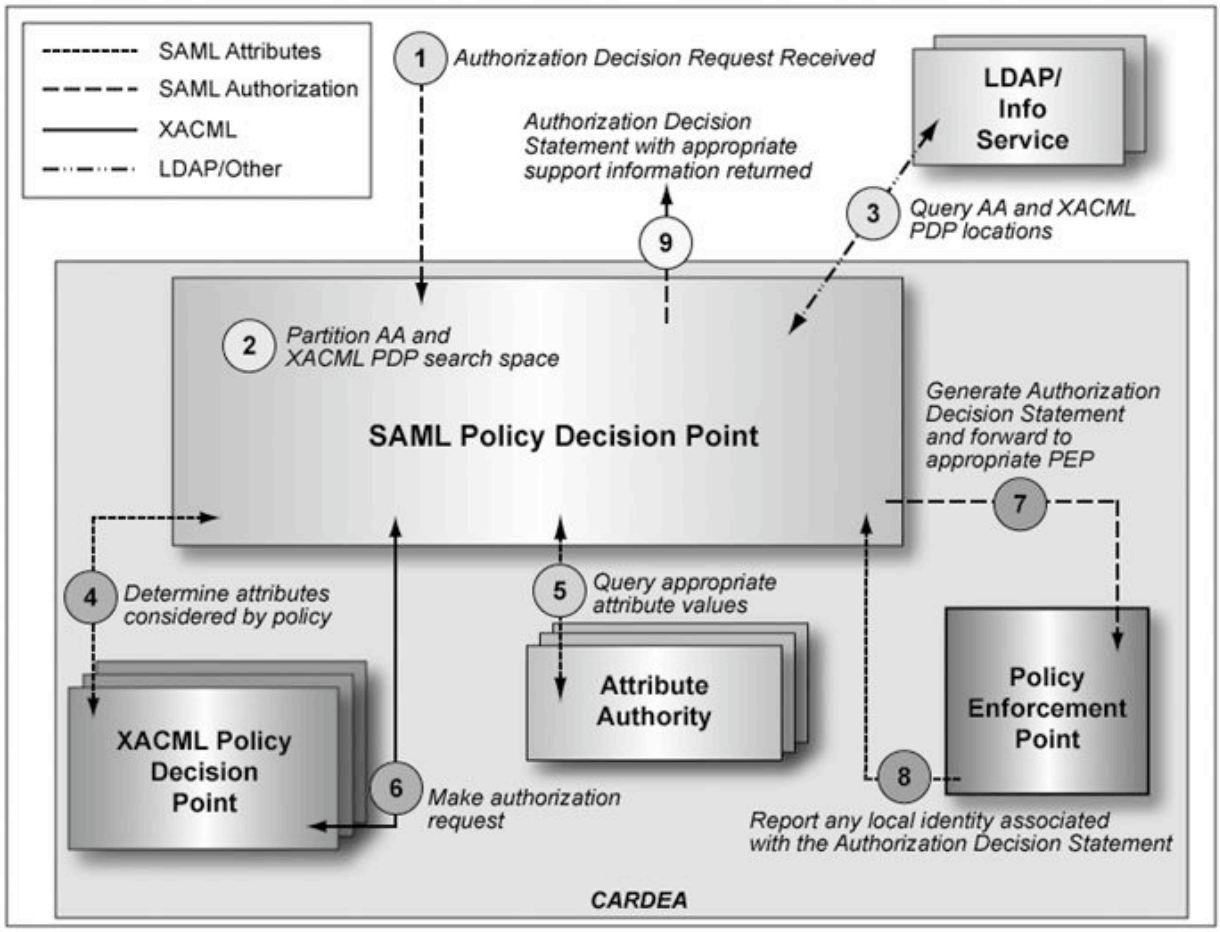
Communication between components is specified directly by the XACML and SAML standards, such as the request and response formats for obtaining information. Although XACML and SAML are transport independent, the initial implementation binds these protocols to the Simple Object Access Protocol (SOAP) v. 1.1 [SOAP]. Support for SOAP-based communication comes from the Java reference implementations of the API for XML messaging [JAXM] and utilizes the Apache Axis [AXIS] architecture as an engine to transmit SOAP messages atop the http and https communication protocols. The Axis engine extracts the raw SAML or XACML construct

from a message payload and forwards to the appropriate endpoint, as configured. From this point, message content is treated as native SAML and XACML and is thus shielded from its method of delivery.

To preserve message integrity, the body of each SOAP message is signed using XMLSig before transmission to the intended recipient. Custom handlers specified for the request and response flows within Axis provide common mechanisms to sign and verify this content of independently of content generation logic. As each message is signed only after processing is complete, the native format of the signed content is opaque to the signing process. Therefore, no dependencies between signature and content must be supported.

The remaining sections examine the current deficiencies in providing distributed authorization and how Cardea identifies functional goals that address those deficiencies. The following sections detail how Cardea provides resolution to these problems by combining the power of XACML and SAML to address those needs. Then, the processing gaps to allow standard functions to interoperate and the manner in which they are filled are investigated. The final section highlights areas that will benefit from additional research and future directions.

Figure 1. Cardea System Architecture



3.1. Research Focus

Cardea research focuses on providing a secure and accountable distributed environment for dynamic resource access across management domains. Not only does this provide higher levels of security and accountability then available under current practices, but also will ultimately allow users to access grid resources where they don't have existing accounts. Further, this approach facilitates the linking of existing authorization component functionality into a complete authorization solution. The ultimate goal in the dynamic resource access area is provision of authorization (can user A access resource Y), dynamic session management (creating/linking accounts as needed for a grid user who does not have an account on the resource) and resource usage reporting capabilities (reporting the usage so that allocations or funds can be adjusted). While each individual step brings the Cardea system closer to offering a complete authorization solution, the following matrix identifies how combinations of intermediate steps can be useful. The following use cases illustrate potential uses of the evolving Cardea system, with the matrix in figure 1 showing what is needed to support it:

A. Providing PIs and resource administrators with the ability to control access to resources. In this scenario, the Cardea system acts as an authorization gatekeeper, collecting data and reaching the appropriate decision.
B. Facilitating immediate emergency access to a set of high-end resources. Any emergency related request could map to existing local emergency identities and thus

no new dynamic enforcement mechanism is required. This could give a set of users instant access to a large number of high-end computational resources to support an emergency analysis.

C. Allowing users and user administrators to query their previous usage to provide a framework for reporting consumption information from each session.
D. Verifying that sufficient allocations exist for a user before a job executes and fails due to running out of allocation in mid-stream.
E. Supporting easier user accounting administration for a grid environment that must support constant change to resource and user populations [HAC01].
F. Enabling cross-grid account-less access, assuming cross grid allocation issues are resolved.

Figure 2. Supported scenarios

Scenario Cases	Authorization	Resource Usage Reporting	Dynamic Session Management
A	X		
B	X		
C		X	
D		X	
E	X		X
F	X	X	X

3.2. Supporting Distributed Authorization

System goals for Cardea attempt to provide new resolution to portion of the overall authorization problem that existing system do not adequately address. The first goal is to model distributed authorization in a way that separates local identity from authorization data. Reaching this goal reduces reliance on preconfigured local identities to enforce authorization decisions. Further, it permits representation and evaluation of access control decisions using strictly the attributes of the entities involved. Finally, it offers a method to provide an authorization enforcement point with sufficient information to manage access control with a variety of enforcement mechanism. The second system goal is supporting distributed authorization while interoperating with existing security infrastructures. This facilitates standard authorization communication between domains by providing a standard representation of global identity and attribute data. Further, it allows each domain to retain control over the mechanisms it uses to provide security services such as authentication and secure communication channels.

3.3. Modeling distributed authorization within Cardea

Cardea departs from the traditional authorization model by providing the support for access control directly using relevant characteristics of the requestor, requested resource and environment rather than pre-existing locally created identities. The system defines access control policies directly in terms of relevant characteristics rather than identities known to possess the appropriate values. Managing authorization information must concentrate on four distinct facets of the selected model. The first facet of this model must consider how the information used by the authorization decision is represented. The second facet selects the

protocol for locating and querying authoritative sources of the information used during the authorization process. The third facet addresses the selection of algorithms for initiating and enforcing the authorization decision. This includes selection of communication partners and the workflow defined. The final facet models the communication paradigms employed. Communication includes mechanisms to digitally sign any shared data, methods to bind the data to a particular transport and the actual communication transport. By focusing on these facets of the authorization model, Cardea does not rely on a specific enforcement approach, authentication mechanism or identity representation. Rather it can leverage any implementation of these functions and provide the way for them to be leveraged in any combination to reach a complete authorization decision.

3.3.1. Identify the information needed for an authorization decision

Cardea does not reduce the amount or type of information needed to make an authorization decision. Rather, it isolates the pieces of data that are considered by the authorization decision and provides the capability to reference that data directly. Anything that can be defined as a characteristic of the subject, the requested resource, the desired action or the current environment may be considered in the authorization decision. Identifying these characteristics and addressing them directly during authorization reduces the need for local identities to represent a set of those characteristics during the authorization process. This model also allows the guarantor of the information to provide data as needed rather than requiring an exchange preceding authorization. The model adopted by Cardea represents these characteristics as SAML assertions that are passed between components. Each component is free to use the assertion data in any capacity it needs, such as transforming it to a different native internal format. However, when communicating the data between components, all characteristics are represented in this common format, regardless of the source or guarantor.

3.3.2. Locating and querying authoritative sources of the information

Separating identity and authorization data divides ownership of authorization data between multiple attribute authorities. These authorities serve assertions only upon request. As the PDP cannot know a priori the subject of a particular request it receives, it requires a protocol by which the PDP can locate the correct AA to query during authorization. Further, the initial content of the request must contain sufficient data to locate any authorities during the authorization process so to request any data that is not packaged directly within the initial request. As each PDP must be capable of evaluating arbitrary access control requests and as each PDP may evaluate a different set of data, the initial request cannot guarantee to contain all data necessary to evaluate a particular authorization request. To locate any missing data, the PDP uses the global identity of the requestor to locate the appropriate authority that can reliably assert any additionally required information. Once located, the SAML specification defines a protocol to request this needed data from the AA.

3.3.3. Initiating and enforcing the authorization decision

Cardea leverages the XACML model for authorization evaluation and SAML for obtaining assertion data used during the evaluation process. Reliance on standards ensures greater interoperability between administrative domains. In fact, if a particular domain does not use XACML to model its authorization policies, it need be only capable of accepting an XACML request and returning a compliant response to participant with other systems. Similarly, Cardea relies upon the SAML protocol to obtain the information necessary to provide to the XACML PDP to make an evaluation decision. Cardea assumes that the SAML PDP that accepts the initial request is responsible for providing the final authorization decision details to the appropriate PEP. If this PEP resides in an external domain, this may require information to flow through a PDP within the external domain to verify trustworthiness of the decision. The SAML PDP depends upon the content of the initial request to determine the correct XACML PDP to evaluate the request. Then, the flow of communication between entities is specified by these relevant standards.

3.3.4. Communication paradigms

Cardea models communication with SAML or XACML without specifying lower level protocols to support communications. Therefore, each component can rely upon existing communication tools and frameworks to manage the details of message transport. In fact, these standards specifically address the use of such protocols through defined bindings. [SAMLB] Further, Cardea defines authorization policies with the XACML policy language. Therefore, any request conforming to standard syntax may be evaluated without any prior knowledge of request or policy specifics. Thus, no control must be ceded to a central mediator to authorize across domains. Further, user administrators need not trust external domains to reliably maintain attribute information on their managed users. Therefore, only the expressiveness of these protocols and their use within an authorization framework limit the control retained by each administrative domain.

4. System Architecture

4.1. System Prerequisites and required configuration

Although the system minimizes the amount of negotiation and configuration that must occur before the authorization framework may be implemented, there are several local items that must be defined according to the standard semantics of XACML and SAML. First, local access control policies must be defined in general terms according to the combined characteristics of salient user-resource combination before they may be enforced by PEP. Additionally, authorities must be populated with appropriate attribute values. Although there is no inherent restriction on how attributes may be maintained or represented internally to its attribute authority, each attribute value must be available to a qualified requester as a SAML Assertion.

4.2. System input

Initial system input must contain the requestor identity, the requested resource and the requested actions. Cardea presumes a SAML AuthorizationDecisionQuery structure packages this information together. Additionally, each AuthorizationDecisionStatement contains a unique

request identifier as per the SAML specification. All system components use only the data contained in this SAML structure or information obtainable using only this data contained within this structure. In the initial system prototype, custom logic generated this initial system input. However, a connector for existing grid toolkits a web service handler or any other custom logic can generate this original SAML AuthorizationDecisionQuery. Within the SAML AuthorizationDecisionQuery, the SubjectConfirmation element contains information on the issuer of the user credential. The SAMLSubject contains the distinguished name from the credential presented with the request. An attribute of the AuthorizationDecisionQuery element holds the Uniform Resource Name (URN) that identifies the requested resource. The specific format for Uniform Resource Names initially used in system is presented in Appendix A. As a particular requestor may wish to use permissions assigned with one or more groups on a given resource, these named groups in which the requestor wishes to participate must be identified within the request. Participation in each named group is then treated as an action to be authorized or denied and is thus represented as a specific SAMLAction. Further, to support charge accounting, the name of the charge group must be provided to the system. Charging to a particular account is also treated as an action to be authorized or denied. Finally, each request may be an initial access request, a follow-up request to query the status of an ongoing process or to terminate a process. Therefore, the specifically requested action requested must be provided as input to the system. Groups, projects and actions are all represented as Uniform Resource Names with a defined namespace.

4.3. System output

The final system output is a SAML AuthorizationDecisionStatement containing the identity of the authorized requestor and the authorized actions including groups, charge account, and authorized resource identity. The identity of the authorized user is represented as a SAMLSubject with the data extracted from the initial AuthorizationDecisionQuery. Each SAML AuthorizationDecisionStatement contains a unique response identifier and a reference to the request identifier for which this response is generated. If any errors were encountered during processing, only a SAMLError structure is returned. Otherwise, the response contains SAMLAssertion structures containing the attribute representations of all authorized actions, including resource access, group participation and charge accounting. The specifically authorized group, charge and resource data are represented as URNs in the same format as they are initially presented in the AuthorizationDecisionQuery. Finally, the XACML PDP decision is also represented within the SAML AuthorizationDecisionStatement.

4.4. SAML architecture overlay XACML architecture

The main component of Cardea is a SAML PDP that accepts an AuthorizationDecisionQuery and responds with AuthorizationDecisionStatement. The SAML standard places no constraints on how a SAML PDP generates query responses. Therefore, Cardea uses XACML functionality to support this function. However, an XACML PDP uses only the information contained directly within a request during evaluation. Within the XACML model, a ContextHandler manages this collection process. Thus, the SAML PDP must act as the ContextHandler to collect and format the needed information into an XACML request to forward to the XACML PDP. Thus, the

SAML PDP also generates any necessary SAMLAttributeQueries and collects the responding SAMLAttributeStatements. Once all attribute statements are received, the SAML PDP transforms the SAMLAttributeStatement data into native XACML format and forwards an appropriate XACML request. The XACML conceptual model assumes that the correct PEP presents the request to the XACML PDP, via the ContextHandler. Therefore, an XACML response contains only the authorization decision and no details from the initial request within response. To ensure that all the information required by the actual PEP is available, Cardea's SAML PDP emulates the ContextHandler to the XACML PDP and preserves all details of the initial request. Then, the XACML PDP decision is incorporated with the preserved context information into the final SAMLAuthorizationDecisionStatement forwarded to the appropriate PEP.

4.5. Decoupling decision and enforcement

A key feature of the Cardea architecture is the division of the conceptual functions of the PDP, AA and PEP into separate components. This allows each site to determine the level of functionality they require. For example, a resource domain need not carry the weight associated with providing user attribute authority functions. By communicating only via the SAML standardized interfaces and according to the selected trust model, the PEP is free to enforce policy decisions from any trusted PDP using its own internal mechanisms. Although each PEP must establish mechanisms to trust an approved set of PDPs, a PEP is not tightly coupled to a specific PDP.

5. Reaching an authorization decision

Each administrative decision within Cardea is processed according to a general algorithm that requires minimal a priori knowledge of participants. The next sub-sections illustrate several critical steps within that authorization process, as handled by Cardea. It specifically highlights communication between distinct system components and how the input and outputs for each component are related.

5.1. Authorization Decision Request Received

In this step, an initial authorization decision request, formatted according to SAML request protocol specifications is accepted by the system. It is important to note that there are no restrictions on the origin of any accepted requested other than is required to enforce local access control policy. For example, an authorization domain may require that any request it processes be authenticated by a trusted source. Any request presenting from an untrusted source would be discarded, regardless of the fact that it could actually be completely processed by the system. All requests in Cardea are processed if they are digitally signed by an identity guaranteed by a trusted authority.

5.2. Partition search space for locating attribute authorities

When an access control decision is required, the authenticated identities of the parties involved are first resolved to their specific authorities that maintain the relevant attribute information. All requesters present a credential to Cardea when requesting an authorization decision. The system

works with existing authentication mechanisms to verify the provided credential. An authority issues each credential. Therefore, the identity of the authority that issued the credential is used to build a query to an information server.

5.3. Query an information service to locate the authoritative AA and PDP locations

After determining the appropriate search parameters by examining the authorization decision request received, Cardea generates queries to an information service to determine necessary location and binding data. These queries use the Java Naming and Directory Interface (JNDI) [JNDI] API to interact with an LDAP server that maintains the appropriate location and binding information for each trusted credential issuer. Cardea places no requirements on the security of interaction with the LDAP server. Each implementation must directly define and support the appropriate means to identity and interaction with trusted information stores. Currently, Cardea assumes location data will be in URL format and needs no authority-specific binding data. As the system matures, sufficient data to locate a portType specification for interaction with an individual authority should be determinable via the information service query.

5.4. Determine attributes considered by controlling policy

Location information for an attribute authority is used to construct a SOAP endpoint that represents an interface to that authority. To minimize the set of attribute assertions presented to the PDP for evaluation, a custom interface was built into the PDP to report the attribute identifiers expected within each request. This interface assumes that the identification of attributes within the XACML policy corresponds with their identification within the attribute servers queried with the SAML protocol. The initial functionality maps resource identifiers to the set of subject attributes required by the policy governing that resource. There is no precedent for this functionality directly within XACML. However, if reporting these attribute identifiers significantly reduces the number of potential attributes that must be collected, it results in a significant efficiency boost over blindly presenting all available attributes to the PDP within each request. Obviously, XACML cannot then specify a format for reporting the set of attributes required by a PDP. Therefore, this information is formatted as SAML attribute statements, permitting a standard interpretation of each result set.

5.5. Query appropriate attribute values

Once locations to obtained attribute information are identified, the relevant attribute values must be queried. Again, XACML specifies only the framework to present a complete set of attribute values to within an authorization request. The standard does not address how to collect the values contained within that set. Thus, SAML Attribute Queries are constructed for the original requester for each attribute required by the controlling PDP. Depending on the initial authorization request, this may require interaction with several distinct attribute authorities, particularly if credentials from several distinct authorities are relevant to the request.

Regardless of the actual attribute authority contacted, the SAML protocol specifies the semantics of extracting the appropriate attribute values. Although not currently enforced, each attribute

authority may also choose to accept or reject requests from an untrusted requester.

5.6. Make authorization request

All attribute information collected within the scope of a single authorization request is preserved to include in the XACML authorization request**.** Once the complete set of requester attributes has been queried, all returned values are formatted as XACML subject attributes. Resource and action attributes are handled in a similar fashion. Cardea employs custom functionality to achieve the necessary integration between XACML and SAML to transform collected attribute assertions to a format recognizable to the XACML PDP. This functionality presumes a correspondence between the attribute identities used in both the XACML and SAML representations of logically equivalent attributes. After populating the request, it is enclosed in a SOAP message destined for the PDP that controls the desired resource. The payload of the response received contains the evaluation decision made by that PDP.

5.7. Generate authorization decision statement to enforce and forward to appropriate PEP

XACML does not define mechanisms to provide detailed information about the access granted in a particular decision. Further, enforcing an authorization decision often requires knowledge of some attributes demonstrated by the requester. For example, if logical group memberships are represented as attributes, the PEP must know in which groups the requester is a valid member. A SAML attribute assertion contains the identity of a particular group where XACML authorization decision specifies membership validity. Therefore, the system bundles the actual authorization decision together with all the attribute values presented to the PDP within the authorization request when interacting with the appropriate PEP. Although not currently incorporated into the final SAML authorization decision statement, evidence used to evaluate the request and conditions attached to the decision may also be presented to the PEP.

5.8. Report any local identity associate with the authorization decision statement

Once the PEP receives a SAML authorization decision statement, its first task is to verify the identity of the SAML PDP that generated the statement. Cardea uses the same generic signature verification handlers added to the Axis [AXIS] request chain as other subsystems. The PEP must define rules that govern how authorization decision statements will be enforced. Initial system design scope concentrated solely on generating the statements to be enforced by a PEP. Several alternative technologies may be used to enforce the design. The only constraint placed on enforcement functionality by Cardea design requires a PEP to report any local identity bound to the authorization decision statement be returned to the initial SAML PDP in the form of a SAML attribute assertion. This constraint facilitates further distribution of the authorization process between distinct yet cooperating PDPs.

6. XACML's role in the authorization process

Initial results demonstrate that XACML fills a critical role within the distributed authorization

framework. Although it does not address all gaps identified in the selected authorization model, XACML's interoperability with other standard protocols provides the mechanism to bridge those gaps. Several of these subjects fall completely outside the scope of the XACML, such as management and retrieval of authorization attributes, or the location of applicable policy decision points. Complimentary technologies, such as SAML or XMLDSig are required to provide this functionality. However, the system must still mediate between the interoperating protocols and develop mechanisms to locate appropriate authorities. Further, although functionality exists to collect attribute information and present it to the appropriate PDP, semantically definitions for common attribute names and legal values must still be negotiated. The remainder of this section examines several such issues that must be addressed and the way that they are addressed in the Cardea system.

6.1. Creation and management of access control policies

XACML provides a mechanism independent representation of access rules that vary in granularity via a standard yet flexible language. This flexibility permits the combination of multiple policies (e.g. from different authoritative parties) into a single applicable policy set to use when making access control decisions for resources in a widely distributed system with overlapping competencies. Further, this mechanism-independent representation of access rules allows a single policy to be applied to heterogeneous resources throughout and across administrative domains. This common representation greatly reduces errors, discrepancies, and auditing complexity. However, creation of actual XACML policies is not a simple task. Further, supporting XACML in heterogeneous environment calls for fully specified data type and function definitions that produce a highly verbose document even if the actual policy rules are trivial. Manual creation of such policies by ordinary users, as required in the PRIMA distributed authority model (see § 9), or by resource administrators, as required in the Cardea system (see §5), is not reasonable. Therefore, additional management tools, such as the introduced PRIMA policy creator, to support policy file management and administration are required.

6.2. Locating the correct PDP

Before an authorization decision can be obtained, an authoritative PDP must be located. This bootstrapping problem is common to any distributed system and not specific to authorization systems based on XACML. Thus, XACML does not provide a standard mechanism to resolve this issue but relies on individual implementations to handle it appropriately to their environment. Initial system implementations either assume that PDP locations are fixed and policy file discovery depends on the requested resource or that each PDP may be located via an information service query to a trusted source. For example, Cardea assumes that a directory service contains the necessary location and binding data for the appropriate PDP. Once a PDP is identified, XACML functionality provides for the location of applicable policy files, including policies to be retrieved from a remote location.

6.3. XACML request preparation and request context management

XACML considers the collection and encoding of attributes used in an authorization system to lie outside its core focus. Further, XACML views attributes as an external form of access control information that must be converted from their native form to be included in an XACML authorization decision request in the form of a request context by a context manager component. XACML does not standardize interactions to retrieve this data for an authorization request. Two distinct approaches have been implemented within the introduced systems to share subject data used for authorization. The first provides a framework by which this information is shared via SAML. The second uses privilege attributes managed by subjects to directly influence the context creation.

The XACML model is based on the authorization pull sequence [RFC2904] and requires the context manager to maintain state information to associate requests that it created with received responses. If another authorization sequence such as the push or the agent sequence [RFC2904] are desired, the contextual information necessary for a PEP to enforce an access decision response from a PDP has to be supplied to the PEP through a supplementary mechanism. Current work on SAML 2.0 proposes to include the original authorization decision request context with an authorization decision response, which would address this issue.

6.4. Encoding of descriptive attributes

Cardea employs SAMLAttributeAssertions to collect and encode attribute data for an authorization decision request. Custom functionality transforms the collected SAMLAttributeAssertions into a valid XACML attribute format. Although specific mappings need not be predefined, the functionality presumes a correspondence between the attribute identities used in the XACML and SAML representations of each logically equivalent attribute. By supporting such transformations, these attributes are available both within the decision and enforcement phases of authorization. Therefore, Cardea augments XACML functionality with SAML functionality to provide this data to all participants in an authorization decision.

7. SAML's role in the authorization process

Like XACML, SAML can also play a central role in the authorization process. SAML clearly defines how to represent attribute information used for access control decisions, a protocol to share attribute information between entities when authorization is distributed, and a protocol to share assertions about authorization decisions. However, SAML does not currently address related issues that impact the architecture of the authorization process. The remainder of this section examines several of these issues and how they fit within the SAML framework.

7.1. Representing attribute information

SAML provides a framework for specifying the format of and parameter naming within an authorization decision request and response. This offers a common basis for heterogeneous entities to make and interpret decision without knowledge of the underlying systems that

implement the decision process. SAML purposely places no restrictions on how a PDP interprets and responds to a request that it receives. Implementers must agree upon expected attribute names, data types and acceptable subject representations for exchanged assertion requests and responses. However, the SAML standard defines constraints on legal representations and how to package this information into assertions. Therefore, using SAML to support communication significantly reduces the amount of agreement required.

7.2. Requesting an authorization decision

SAML provides a framework that answers many of the design questions that any system architect needs to answer when building an interoperable authorization system. [COHEN] Specifically, it provides a blueprint for the communication necessary to transform an assertion request into an assertion statement. However, it places no requirements on the methods and functionality by which a particular authority implements these transformations. With Cardea, this process requires contact with suitable authorities, collection of support information and evaluation of relevant access control policies. Although this bootstrap problem of locating the proper authorities is not peculiar to SAML based systems, it must still be addressed. Cardea uses well-known grid information services to locate the appropriate authorities whereas alternative solutions may rely on UDDI, LDAP, RDBMS or OGSA discover technologies. Fortunately, SAML provides a way to model data items needed during the authorization process: as attributes that may be queried from an attribute authority and specified in the form of assertions. Therefore, each PDP needs to develop only the logic for location mechanisms and the SAML functionality to collect the information it requires. Further, XACML functionality specifies a standard mechanism to format access control policies and evaluate requests according to a specified policy. Therefore, overlaying SAML functionality on XACML functionality provides a clear roadmap to deterministically transforming an initial SAML AuthorizationDecisionQuery into a SAML AuthorizationDecisionStatement.

8. Related Work

Grid researchers approach the problem of authorization in a grid environment from varying directions. Each approach must consider issues such as access control policy representation, policy enforcement, delegation of rights and user management. There are several current systems that implement an alternative solution to the problem of distributed authorization. These systems include: the Community Authorization Service (CAS) [PEA01] from ISI, the VOMS [ALF01] system from the European Data Grid, Akenti [MUD01], the PRIMA [LOR01] system research from Virginia Tech University, and the PERMIS system [CHA01]. Each system models the authorization problem uniquely, according to their selected focus available technology.

CAS leverages the Globus GSI, to allow resource providers to specify course-grained access control policies in terms of communities as a whole. Resource administrators can thus delegate fine-grained access control policy management to the community itself. Therefore, resource providers maintain ultimate authority over their resources but are spared day-to-day policy

administration. CAS requires each community member to authenticate with the community credential when accessing a resource with community rights. Further, although administration is partitioned between community and resource administrators, knowledge of resources, user identities, community access rights and group memberships must be established before authorization may occur.

Alternatively, the PERMIS system focuses on both authentication and authorization with the use of identity and attribute certificates to represent both the mutable and immutable model of a user. While implemented with XML and Java technologies, the access control API as well as mechanisms to allocate privileges are currently proprietary. Further, the reliance on a pull model to collect attributes and the X.509 attribute certificate standard to represent attribute values represent assumptions that are not necessarily valid across heterogeneous environments.
Similarly to the CAS, the Virtual Organization Management System (VOMS) also performs authorizations according to community membership privileges. A VOMS server encodes this information directly into non-critical certificate extensions that may be presented to the resource. Thus, the relevant authorization information is packaged directly with the identity information used for authentication and pushed to the requesting resource. The system assumes that group and role information is reported in a format natively understandable to the authorized resource. Therefore, although the system distributes management responsibilities between resource and VO administration, it still presupposes this information is known before the authorization decision may occur.

Akenti extends the CAS/VOMS community model to support multiple stakeholders that may impose use conditions on a particular access control request. Akenti provides a way to express and to enforce an access control policy without requiring a central enforcer and administrative authority. The system's architecture is intended to provide scalable security services in highly distributed network environments. Attribute and use case conditions are encoded in digitally signed certificates that are collected during the authorization decision process (pull model) and evaluated according to the Akenti policy language. Although the pull model for collecting data facilitates system distribution, the proprietary policy language developed for Akenti requires that any participant use specialized, rather than standard, data formats and APIs.

Finally, the PRIMA system provides mechanisms that focus on the management and the enforcement of fine-grained access rights. The solution employs standard attribute certificates to bind rights to users (or their surrogates) and enables the high level management of such fine grained privileges which may be freely delegated, traded, and combined. Enforcement is provided by POSIX operating systems extensions that extend standard file permissions and regulate resource usage through access control lists. These extensions are available for common platforms and fully support legacy services. In combination, our privilege management and enforcement mechanisms are compatible with and enable the usage of fine-grained rights, leverage other work in the grid computing and security communities, reduce administrative costs to resource providers, enable ad-hoc collaboration through incremental trust relationships and can be used to provide improved security service to long-lived communities.

9. Future Directions

Work on the Cardea system will proceed in two complimentary directions. First, work will be performed on integrating basic system functionality into the existing NASA computing infrastructure. Integrating the functionality requires both interfaces to existing systems as well as modification on policies and processes to support the framework. Further, current features of the system will be expanded to move from an authorization prototype towards a robust accounting solution. This requires additional functionality that considers dynamic account management and allocation management. The specific tasks to support these dual goals are outlined below.

- Define NASA Ames access control policies using the XACML specification. Perform necessary infrastructure support work to ensure logic represented in the policies complies with governing NASA security policy. Determine and implement necessary security for the actual policy files.
- Integrate Cardea with the Open Grid Services Architecture (OGSA) [FOS04]. This requires definition of a portType, according to the grid service specification, and leveraging Globus credential and authentication mechanisms.
- Build an attribute authority to support IPG credentialed entities. Standard attribute definitions for each supported attribute within the authority must be established. This authority must integrate with existing NAS user management systems that already management attribute values. Finally, the appropriate support framework for secure communications with the attribute authority must be developed.
- Develop a dynamic session management system for authorization policy enforcement that relies only on authorization decision information rather than a unique identity for each potential user. This subsystem will manage any local identities created to support an authorization decision. In fact, local identities need be created only at the discretion of the resource administrator, rather than as the de-facto authorization enforcer [HAC02].
- Provide a framework to report consumption information from a dynamic session to the actual administrative domain of the requesting user. The framework will report consumption information with the Global Grid Forum (GGF) Usage Record XML format [GGF01, GGF02] so that all information needed to charge the appropriate user within that domain is available.

Any dynamic resource access system must also consider resource usage and allocations. Therefore, work to provide basic resource usage and allocation functionality will proceed in parallel to authorization work. Resource usage reporting will allow users and user administrators to query their previous usage. Such reporting creates the basis for verifying that sufficient allocations exist for a user before a job even executes. Provide a framework to report consumption information from a user's session to the hosting or administrative domain of the requesting user. The framework will report consumption information with the GGF Usage Record XML format so that all information needed to charge the appropriate user within that domain is available.

Finally, coupling the new authorization function with dynamic session management capabilities will support easier user accounting administration for a grid environment that supports constant

change to resource and user populations. Researchers at other institutions are currently developing dynamic session management systems for authorization policy enforcement that relies only on authorization decision information rather than a unique identity for each potential user that will manage any local identities created to support an independently generated authorization decision. This ability, together with the resource accounting features outlined above provides the ability for cross-grid account-less access, assuming cross grid allocation issues are resolved. As this capability is still emergent, research this year will focus on investigating and evaluating solutions and their fit with existing authorization portions.

Acknowledgements
This work was supported by NASA Ames Research Center under Contract Number DTTS59-99-D-00437/TO#A61812D with AMTI/CSC.

References

[ALF01] R. Alfieri et al: *VOMS, An Authorisation System for Virtual Organization*. First European Across Grids Conference, Santiago de Compostela, Feb. 13, 2003

[AXIS] http://xml.apache.org/axis, visited 2003-07-14

[CHA01] Chadwick, D., and A. Otenko: *The Permis X.509 Role Based Privilege Management Infrastructure.* SACMAT 2002 Conference Proceedings, ACM Press, NY, pp. 135 – 140

[COHEN] Cohen, F.: Debunking SAML myths and misunderstanding. http://www-106.ibm.com/developerworks/xml/library/x-samlmyth.html, Jul. 8, 2003

[FAWC]Patrizio, A.: *SAML advances single sign-on prospects.* http://www.fawcette.com/xmlmag/2002_03/magazine/departments/marketscan/SAML/

[FOS01] Foster, I., Kesselman, C., and S. Tuecke: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. International Journal of Supercomputer Applications 15(3); p.200-202. 2001

[FOS02] Foster, I., Kesselman, C., Tsudik, G., and S. Tuecke: *A Security Architecture for Computational Grids.* ACM Conference Proceedings, Computers and Security, ACM Press, p. 83-91, 1998

[FOS03] Foster, I and C. Kesselman: *Globus: A Toolkit Based Grid Architecture.* The Grid, Blueprint for a Future Computing Infrastructure, Morgan Kaufmann, San Fracisco, p. 259-278, 1999

[FOS04] Foster, I., Kesselman, C., Nick, J., and S. Tuecke: *Grid Services for Distributed System Integration.* Computer 35(6), 2002.

[GGF01] The Global Grid Forum – http://www.gridforum.org, visited 2003-07-14

[GGF02] The Global Grid Forum Usage Record Working Group – https://forge.gridforum.org/projects/ur-wg/, visited 2003-07-14

[HAC01] Hacker, T. and B. Athey: *Account Allocations on the Grid.* DRAFT. Online at http://www.nas.nasa.gov/~thigpen/accounttemplates.pdf, 2002

[HAC02] Hacker, T. and W. Thigpen: *Distributed Accounting on the Grid.* Global Grid Forum

Account Management Working Group, October 1999

[HUM01] Humphrey, M., Knabe F., Ferrari, A. and A. Grimshaw: *Accountability and Control of Process Creation in Metasystems.* Proceedings of the 2000 Network and Distributed System Security Symposium (NDSS2000), February 2000

[IPG01] The Information Power Grid - http://www.ipg.nasa.gov, visited 2003-07-14

[JAVA] The Java Language, http://java.sun.com, visited 2003-08-01

[JAXM] The Java API for XML Messaging, http://java.sun.com/xml/jaxm/, visited 2003-07-31

[JNDI] The Java Naming and Directory Interface, http://java.sun.com/products, jndi, visited 2003-08-01

[JOH01] Johnston, W., Mudumbai, S and M. Thompson: *Authorization and Attribute Certificates for Widely Distributed Access Control.* IEEE 7th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1998

[LOR01] Lorch, M., and D. Kafura: *Supporting Secure Ad-hoc Collaboration in Grid Environments.* Proc. 3rd Int. Workshop on Grid Computing, November 2002

[MIR01] Miroslaw, K. Meyer, N. and P. Wolnieewicz: *Simplifying Administration and Management Processes in the Polish National Cluster.* Poznan Supercomputing Center, 2001

[MUD01] S. Mudumbai et al: *Akenti A Distributed Access Control System.* Online at http://www-itg.lbl.gov/security/publications.html

[OASIS] Oasis, http://www.oasis-open.org/index.php, visited 2003-08-01

[PEA02] L. Pearlman et al: *A Community Authorization Service for Group Collaboration*. IEEE Workshop on Policies for Distributed Systems and Networks, 2002

[RFC2459] R. Housley et al: Internet X.509 Public Key Infrastructure Certificate and CRL Profile. IETF RFC, January 1999

[RFC2904] J. Vollbrecht et al: AAA Framework. IETF RFC, August 2000

[RFC3281] S. Farrell, R. Housley: An Internet Attribute Certificate Profile for Authorization. IETF RFC, April 2002

[SAML] P. Hallam-Baker et al: Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML). Oasis Standard, November 5th, 2002

[SAMLB] P. Mishra et al: Bindings and Profiles for the OASIS Security Assertion Markup Language (SAML). Oasis Standard, November 5th, 2002

[SOAP] Don Box et al: *Simple Object Access Protocol (SOAP) 1.1.* World Wide Web Consortium Note, May 2000

[TUE01] S. Tuecke et al: Internet X.509 Public Key Infrastructure Proxy Certificate Profile. IETF draft, 2001

[W3C] World Wide Web Consortium, http://www.w3c.org, visited 2003-08-01

[WS01] Webservices.org article: *XACML ratified as OASIS Open Standard.* http://webservices.org/index.php/article/articleview/909/1/65/, Feb. 18, 2003

[WSDL] E. Christensen et al: *Web Service Description Language,* W3C Note, Mar. 15, 2001

[XACML] S. Godik et al: *eXtensible Access Control Markup Language (XACML) Version 1.0.* OASIS Standard, February 18th, 2003

[XML] T. Bray et al: *Extensible Markup Language Version 1.0*. W3C Recommendation, Oct. 6, 2000

[XMLD] Mark Bartel et al: *XML Signature Syntax and Processing*. World Wide Web Consortium Recommendation, Feb. 2002

10. Appendix A - Attribute Definitions and Identifiers

Each entity addressed or examined within the Cardea system must be represented with the SAML language syntax. This section outlines the name, type and when necessary value decisions used.

10.1. Resources

When representing resource identity with a SAMLAuthorizationDecisionStatement or a SAMLAuthorizationDecisionQuery, the following format is used:
Namespace: urn:das:ipg:resource:1.0
Legal value: FQDN of the requested resource
Example:
To request access to the machine cafe210.nas.nasa.gov, the identifier used would be:
urn:das:ipg:resource:1.0:cafe210.nas.nasa.gov
When representing resource identity within the request to the XACML PDP, the following attribute formats are followed:
Attribute Identifier: urn:oasis:names:tc:xacml:1.0:resource:resource-id
Value: FQDN of the requested resource

10.2. Requester identity

The SAML subject containing the requestor identity extracts the distinguished name from the presented credential, formatted as a globus distinguished name [GLOB], and inserts that as the value of the subject. Each SAML subject must also identify the format of the subject identity, using a URN identifier. This identifier is specified by the SAML standard as:
urn:oasis:names:tc:SAML:1.1:nameid-format:X509SubjectName

10.3. Actions

Namespace: urn:ipg:names:tc:DAS:1.0:action:login
Legal value: Map, Status, Touch, Unmap
Example:
To identify the initial mapping action, the URN would be urn:ipg:names:tc:DAS:1.0:action:login:Map
To identify an action to query the status of a process, the URN would be urn:ipg:names:tc:DAS:1.0:action:login:Touch
Any URN that specifies and action not represented in the legal values must be rejected as an unknown action.

10.4. Groups

Namespace: urn:ipg:names:tc:DAS:1.0:action:group
Value: name of the requested group.
Example:
To identify the ipg-development group, the URN would be urn:ipg:names:tc:DAS:1.0:action:group:ipg-development

10.5. Projects

Namespace: urn:ipg:names:tc:DAS:1.0:action:charge
Value: name of the charge group.

Example:
To identify the ipg-project charge group, the URN would be urn:ipg:names:tc:DAS:1.0:action:charge:ipg-project

10.6. X500 Attributes

X.500 Attributes are used to represent standard pieces of data about users, either within an XACML policy file, or within a SAMLAttributeQuery, a SAMLAttributeStatement or an XACML request. Each of these attributes must be uniquely identified within those structures. Therefore, the standard namespace and short names for those attributes are used to create a unique identifier for those attributes. Any X.500 attribute can be identified by appending the short name from the rfc2256 specification [RFC2256] to the X.500 namespace, separated with a hash mark (#)

X.500 Namespace: http://www.ietf.org/rfc/rfc2256.txt

Example:
To identify the citizenship attribute, the generated URN would be http://www.ietf.org/rfc/rfc2256.txt#c

11. Appendix B – Sample Policy File

This appendix contains a simplified XACML policy file that controls access to a resource according to the value of the citizenship attribute provided in a presented XACML request context.

```
<?xml version=”1.0” encoding=”UTF-8”?>
<Policy PolicyId=”TuringPolicy”
RuleCombiningAlgId=”urn:oasis:names:tc:xacml:1.0:rule-combining-algorithm:first-
applicable”>
<Target>
<Subjects>
<AnySubject/>
</Subjects>
<Resources>
<AnyResource/>
</Resources>
<Actions>
<AnyAction/>
</Actions>
</Target>
<!-- Rule to see if we should allow the Subject to login -->
<Rule Effect=”Permit” RuleId=”USCitizenRule”>
<Description>This rule verifies that all users wishing to perform any account managment (Map,
Unmap, Query) satisfies the characteristic, using LDAP naming/format of attributes,  that
http://www.ietf.org/rfc/rfc2256.txt#c  is  ‘US’</Description>
<Target>
<Subjects>
<SubjectMatch MatchId=”urn:oasis:names:tc:xacml:1.0:function:string-equal”>
<AttributeValue
DataType=”http://www.w3.org/2001/XMLSchema#string”>US</AttributeValue>
<SubjectAttributeDesignator    AttributeId=”http://www.ietf.org/rfc/rfc2256.txt#c”
DataType=”http://www.w3.org/2001/XMLSchema#string”/>
</SubjectMatch>
</Subjects>
<Resources>
<AnyResource/>
</Resources>
<Actions>
<AnyAction/>
</Actions>
</Target>
</Rule>
<!-- A final, “fall-through” Rule that always Denies -->
<Rule Effect=”Deny” RuleId=”FinalRule”/>
</Policy>
```

12. Appendix C – A SAMLAuthorizationDecisionQuery and the generated XACML request

This appendix shows the data contained within a SAML request (in debug format) and the XACML request context that Cardea generates from the request.

********** *CONTEXT INFO* ********

RequestID: 6a98b92c-8666-441a-ac14-7104fe04688c
Query:AuthorizationDecisionQuery
Provider:CN=Certificate Manager, OU=Ames Research Center, O=National Aeronautics and Space Administration, O=Grid
Action requested:
Namespace: urn:ipg:names:tc:DAS:1.0:action:login
ActionData: Map
Action requested:
Namespace: urn:ipg:names:tc:DAS:1.0:action:group
ActionData: ipd-dev
Action requested:
Namespace: urn:ipg:names:tc:DAS:1.0:action:group
ActionData: non-existant-group

The resulting XACML request:

<Request>

<Subject SubjectCategory="urn:oasis:names:tc:xacml:1.0:subject-category:access-subject">
<Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id" DataType="http://www.w3.org/2001/XMLSchema#string" IssueInstant="2003-08-07T11:21:59.481000000-07:00"><AttributeValue>CN=Rebekah Lepro,OU=Ames Research Center,O=National Aeronautics and Space Administration,O=Grid</AttributeValue></Attribute>
<Attribute AttributeId="http://www.ietf.org/rfc/rfc2256.txt#o" DataType="http://www.w3.org/2001/XMLSchema#string" IssueInstant="2003-08-07T11:21:59.481000000-07:00"><AttributeValue>National Aeronautics and Space Administration</AttributeValue></Attribute>
<Attribute AttributeId="http://www.ietf.org/rfc/rfc2256.txt#c" DataType="http://www.w3.org/2001/XMLSchema#string" IssueInstant="2003-08-07T11:21:59.480000000-07:00"><AttributeValue>US</AttributeValue></Attribute>
</Subject>
<Resource>
<Attribute AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id" DataType="http://www.w3.org/2001/XMLSchema#anyURI" IssueInstant="2003-08-07T11:21:59.483000000-07:00"><AttributeValue>urn:das:ipg:resource:1.0:turing.nas.nasa.gov</AttributeValue></Attribute>

</Resource>
<Action>
<Attribute AttributeId=”urn:oasis:names:tc:xacml:1.0:action:action-id” DataType=”http://www.w3.org/2001/XMLSchema#anyURI” IssueInstant=”2003-08-07T11:21:59.483000000-07:00”><AttributeValue>urn:ipg:names:tc:DAS:1.0:action:login:Map</AttributeValue></Attribute>
</Action>
</Request>